EE-224 Project
Modelling RAM using digital electronics

Name : Ram Prakash Roll No. : 210260042
Name : Siddharth Stephen Roll No. : 210260049

Final Project Executive Summary:

Our goal was to model Random Access Memory using digital electronics.

We finally ended up building a 16 bit memory in the form 4 blocks of 4 bit memory. We designed a machinery of addressing these 4 blocks using a 2 bit address allotted to each memory block. A parallel input using 4 data lines is stored in a 4 bit memory block, using a D register. Data is then temporary stored there, and can be changed easily. Then data is transferred serially from this temporary memory block to those 4 bits memory blocks according to address chosen. By using this method we achieved higher speed and accuracy (by avoiding direct exposure of memory bits to inputting data) compared to traditional D-register memory blocks where data was inputted serially, where synchronisation between clock and data (inputted manually by the user) had to be taken care of. Finally, Data stored in D-register was displayed on digital display in parallel fashion. In a nutshell, we implemented :

1. Address allocation to 4 memory blocks.
2. Parallel input stored in a temporary memory, so as to avoid direct exposure of inputting data to 16 bit memory blocks.
3. Serial data transfer from temporary memory block to 4 bits memory blocks according to address allocated.
4. Delete and Redo feature implemented using temporary memory block (Cash Memory).
5. Parallel output corresponding to each memory block displayed on digital display (shows decimal output corresponding to binary output).

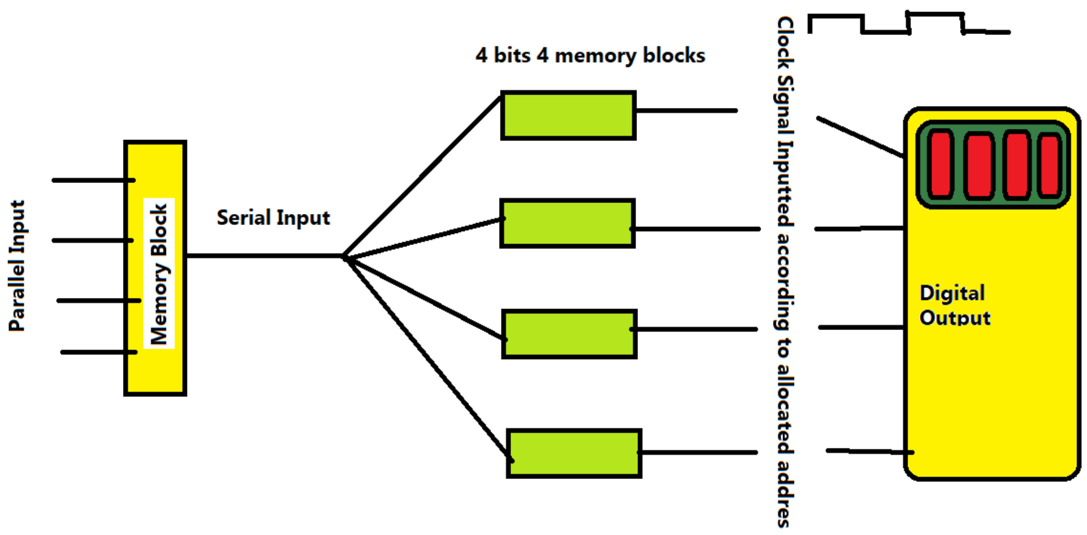


Figure 1: Rough Circuit Diagram

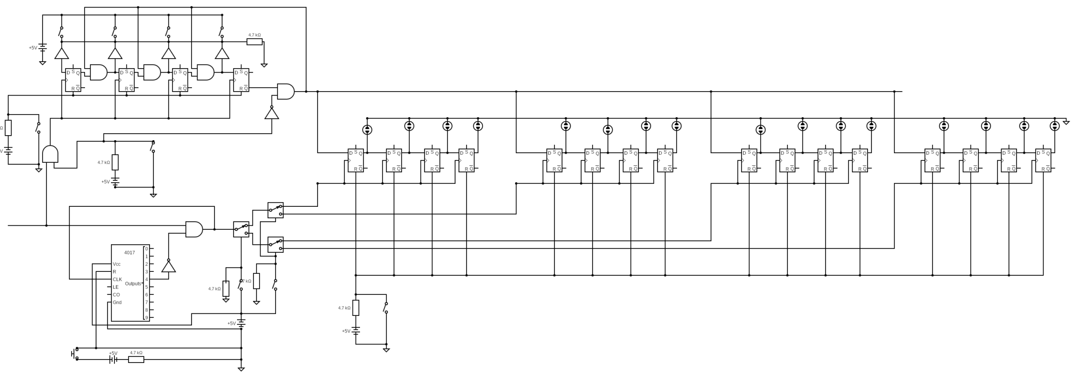


Figure 2: Final Circuit Diagram

Final Project details:

As I have already mentioned what features we have implemented in our final project. This sections contains information about theoretical design, circuit diagrams, it's application in current technologies, etc.

— — — — — — — — — — — — — — — — — — — —

Theoretical design

As I have already mentioned our aim was to model Random Access Memory using Digital Electronics. We used 4bit 5 D-registers representing four 4 bit variables and one 4 bit for storing input temporarily, a de-multiplexer for allocating address to clock signals of these D-registers and tristate logic buffers used to generate the three stages i.e. Input, Storage and Unconnected. We will be giving parallel input to a D-register and from this register we will give serial input to other registers with ultra high speeds. This method won't rely on synchronisation of clock with manual input. Whole algorithm:

1. De-multiplexer's output is fed to one end of AND gate, other end being the Clock Signal. Output of this AND is fed to Clock signals of D-registers. This way we achieved addressing of D-registers.

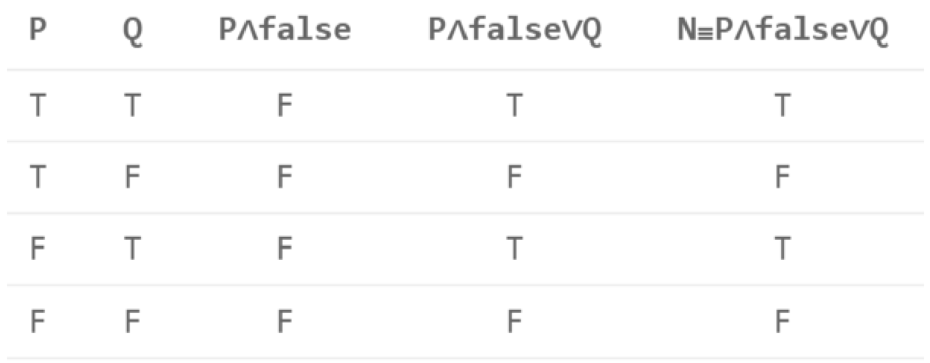
2. For generating 4 clock cycles, we used FSM along with Counter IC,which is explained later. We need FSM here to resolve a very serious issue, i.e. user can press the reset switch anytime. When clock was high and user presses the switch, Counter doesn't count the first signal, i.e. from zero to high state, but D-register does count it. Therefore we wanted to include a FSM which solves the issue. Resolution of this problem has been explained after this section.

3. For accomplishing tri-state logic we used a buffer IC. Using this we implemented two states of temporary memory block D-register. One of the two states are 'updating' mode' in which whatever changes user makes in input will be reflected in D-register after one clock cycle. Second state is 'Storage mode', i.e. no further changes can be made to data stored in D-register. In this mode, data is ready to be transferred to the 4 memory blocks.

4. Data is then transferred serially to the memory block whose address has been chosen as soon as we press the reset switch of counter (which stops after 4 clock cycles).

5. Data is then displayed on digital display.

We were required to use an FSM in our project. A very simplified one actually - The idea is to wait until after the updating of one state of the finite machine, giving time for the counter to reset before starting the clock. This ensured that the counter measured the exact number of clock cycles received by the memory



P	Q	P∧false	P∧false∨Q	N≡P∧false∨Q
T	T	F	T	T
T	F	F	F	F
F	T	F	T	T
F	F	F	F	F

Figure 3: FSM Truth Table

blocks - four in our case. S_1 is the state when Clock is high. S_2 is the state when Clock is low. Q here is negation of Q_C and it is the input to the FSM. FSM only depends on Q actually. Next state is simply negation of Q. Whenever Q is high , next state will be S_2 and that means that clock will be made available to the circuit after one cycle of the FSM, which is half the time of the cycle of the clock. P represents the present state of the FSM and N is the next state of the FSM.

$$\begin{tikzcd}
S_1;1 \arrow[loop left, "Q=0"] \arrow[rrr, bend left, "Q=1"] & & & S_2;0 \arrow[loop right, "Q=1"] \arrow[lll, bend left, "Q=0"]
\end{tikzcd}$$

Components required for building this circuit are:

- 4+1 D-register (74LS174 IC) : 1 For temporary + 4 permanent memory
- Counter IC (74LS90) : For generating 4 clock cycles
- Demultiplexer (74154N IC) : For addressing the D-registers
- 2 Buffer IC's (74LS244 IC) : For modelling different states, namely, updating a.k.a inputting mode and Storage mode.
- AND gates (74LS08 IC) : For modelling FSM.

- NOT gates (74LS04 IC) : For modelling FSM.
- LED Bulbs : For showing outputs.
- Switches : For IC reset and data input purposes.

— — — — — — — — — — — — — — — — — — — —

Experimental implementations

Here we present how will we demosntrate our project. Here is the point wise summary of our demonstration:

- Step 1 : Turn on Vcc and Clock Signal. Then reset all D-registers simultaneously by pressing CLR switch.
- Step 2 : We have to switch to updating mode(whatever we give as input will be simultaneously stored) of temporary memory to first store the data in temporary memory. Then set the values of input data. Once you are sure about the data that has to inputted follow step 3.
- Step 3 : We then switch to storage mode(data stored can no longer be changed by changing inputs). In this mode data has been stored in temporary memory block. Data is ready to be transferred to those 4 memory blocks.
- Step 4 : We then select address of the D-register in which we want to input data. Then we press the reset switch of counter after which data storage in selected D-register starts.
- Step 5 : Once, your data has been stored, you can over-write your data by following previous steps, without afftecting other memory blocks. You can restore your data (temporarily) if somehow your pressed reset switch.

Connection with Computer Science

Our motivation, that we have already mentioned in our previous report; was to understand and model how RAM actually works. Keeping this in mind, our goal was to model an efficient, accurate and faster memory handling RAM. Accordingly, we included features of Redo and Delete also. Nevertheless, it's pretty much evident how RAM forms the basis of whole of Computer Technologies. We targeted the very basics of computer sciences.

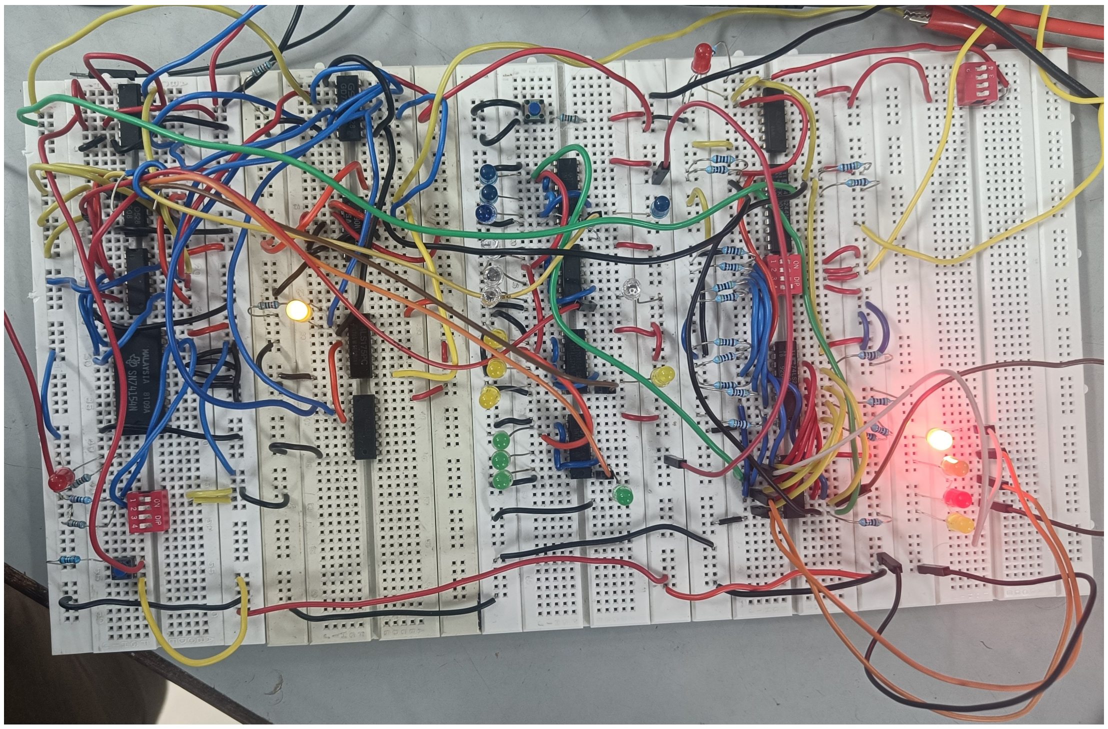


Figure 4: Final circuit

Appendix

We started out with modelling RAM just with D-registers, Logic gates and counters. We ended up getting a whole mess out of this. Then we came to know about de-multiplexers which helped us accomplish our task of addressing the D-registers. Next, the biggest problem we faced was to generate 4 clock cycles, given that user may press the reset button anytime during the clock cycle. It took us whole 2 weeks to figure out this issue. Finally, Siddharth came out with an innovative solution to this problem using FSM. In our last week, after we have tested the baisc stuffs, Siddharth initiated to include tristate logic buffers to increase the speed and accuracy of memory storing capacities of this RAM. We finally agreed upon to take this risk in the last week, and we ended up completing our task on Sunday evening, i.e. on 9th April,2023. Link for our LTSpice Simulation and

Demo Video : https://drive.google.com/drive/folders/1J2fb7oGgYickVMbCk–eQNpVMAGxQ3cf